Error Surface Generation Techniques for Appearance-based Stabilisation of an Intelligent Kite Aerial Photography Platform (iKAPP)

Hosseini, H.; Neal, M. J.; Labrosse, Frédéric

Published in:
TAROS 2008: Towards Autonomous Robotic Systems 2008

Publication date:
2008

Citation for published version (APA):
Hosseini, H., Neal, M. J., & Labrosse, F. (2008). Error Surface Generation Techniques for Appearance-based Stabilisation of an Intelligent Kite Aerial Photography Platform (iKAPP). In S. Ramamoorthy, & G. M. Hayes (Eds.), *TAROS 2008: Towards Autonomous Robotic Systems 2008* (pp. 163-170). Edinburgh University Press.

Error Surface Generation Techniques for Appearance-based Stabilisation of an Intelligent Kite Aerial Photography Platform (iKAPP)

Heidar Hosseini, Mark Neal, Frédéric Labrosse
Department of Computer Science, Aberystwyth University, Ceredigion,
Wales, UK, SY23 3DB
{hhh05, mjn, ffl}@aber.ac.uk

***Abstract*— This paper presents results obtained whilst attempting to use only image comparison techniques to actively stabilise a pan-tilt camera mounted on an aerial platform. We use the camera to take consecutive images, calculate the difference and pass the output to the control system in order to move the camera to its original location. The goal is to stabilise the camera's view no matter how far we move the camera. Image processing constraints interact with the control system constraints. In this paper we present various techniques and experiments that are focused on the selection of pixels to be used in generating suitable error surfaces to allow the camera to be returned to its original view. Patch positioning and whole image sampling are the two techniques examined.**

I. Introduction

Aerial photography is often done from light-weight platforms such as radio-controlled aircraft or kites. The advantages of such platforms are that they can be transported to remote places and deployed rapidly. However, they also suffer dramatically from difficult wind conditions, resulting in their on-board camera moving in difficult to control ways. Some stabilisation can be achieved using gyroscope-based systems, which maintain the pose of the camera rather than making sure it always points at the feature of interest.

In this project, we are interested in performing stabilisation using the images themselves, ensuring that they always contain a specified area on the ground. More specifically, patches of images are compared to the current image which is "moved" using an active pan-tilt platform to maintain the correct aim. In this paper we present different schemes to select these patches based on different strategies.

II. Error Surface

A. What are "Error Surfaces" and how do we use them?

To calculate the distance between two images, we capture selected pixels from a reference image and calculate the Euclidean distance between this selection and the corresponding pixels from the current image. In order to generate the error surface we repeat the same process over the entire image by moving the patch taken from the reference over every possible point on the current image but bearing in mind that the reference patch edges shouldn't cross the current image border. The distances calculated at each location can be used to represent the height of the "error surface" at each point. The distance between two patches is calculated as follows:

$$d(I_i,I_j)=\sqrt{\sum_{k=1}^{h\times w}\sum_{l=1}^{c}(I_j(k,l)-(I_i(k,l))^2},$$

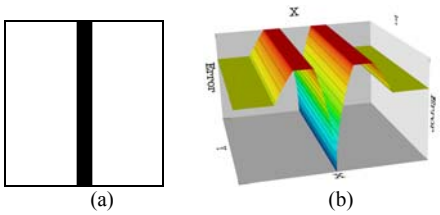
where h and w are the patch's height and width and k represents the current pixel. $I_i(k,l)$ and $I_j(k,l)$ are the l th colour component of the k th pixel of images I_i and I_j respectively [5].

In order to stabilise the camera, we need to find the best match between the patches under consideration. This can be represented as finding the minimum on the error surface generated using this method.

B. What image features generate what error surface?

For this work we use the RGB colour space, although we are aware that other colour spaces may have different and possibly beneficial properties.

The shape and location of the patches used in this process can dramatically affect the outcome. For example a reference patch drawn horizontally or vertically across the image may match well with a number of similar regions in the image, which can cause the error surface to have local minima and/or an indistinct global minimum. For example, Fig. 1(a) shows an image consisting of a vertical line drawn in the middle. Using a *Central* patch (see Section IV) causes an indistinct global minimum that leads to ambiguity in finding the optimal camera location, Fig. 1(b).



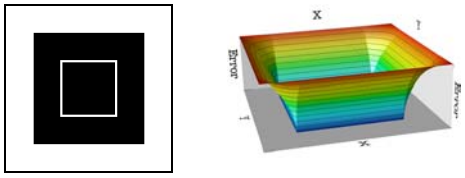
(a) (b)

Fig. 1: Error surface (b) from an image with a vertical line crossing the middle (a)

When generating the error surface, the selected patch also affects the resultant error surface. For example the box in the middle of the black area in Fig. 2(a) shows what the central patch is targeting. This interacts with the image to generate an error surface with a large ambiguous global minimum.

The error surface in Fig. 2 was generated by applying the comparison process using the captured patch (central box) over the entire image (Fig. 2(b)). This error surface shows

the large number of minimal points we encounter by targeting a central patch in the above image.



(a) (b)

Fig. 2: Error surface (b) from an image with a black box in the middle (a)

C. *Error surfaces in real world images.*

The above comparisons were produced based on some artificial images but, is it the same in real images? The above comparisons of artificial images show clearly that the final outcome depends on the patches we target as well as the image that we process.

In real images small patch movements may cause large changes in the image's characteristics and may result in dramatic changes in error value and therefore to very steep error surfaces. The challenge is therefore to find a sampling strategy that minimises this tendency whilst maximising the likelihood of generating a well-defined and unambiguous global minimum. Thus, establishing what regions to target when dealing with real world images will be the key to generating well-behaved error surfaces. In order to test these ideas we have applied these techniques to some real world images.

Repeating colour patterns, large regions of constant colour and complex textures are all features which need to be assessed and thus images containing these features have been selected for testing (see Fig. 4).

III. Control Algorithm ("P" Algorithm)

In order to meaningfully assess the utility of the error surfaces we need a control system that can use them to find the minimum. There are a variety of control algorithms that can be used to control systems such as that presented here. PID (Proportional, Integral, and Derivative) [3] is one of the simplest and best understood and a slightly modified proportional controller will be used in this work. The positioning system itself is beyond the scope of this paper although the authors have designed, built and tested such a system which has sufficient speed, precision and resolution for this application.

Our "P" algorithm enhances the controller movement by increasing the P gain iteratively when the control system fails to generate a movement. This is a simple low-overhead heuristic to avoid getting stuck on flat regions of the error surface. The P gain is reset to 1 after each movement. This algorithm is not intended to be used in a final control system, but is simply used to assess the error surfaces generated in this work.

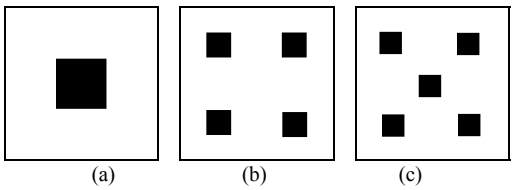
IV. Strategies for Patch Positioning

What and where to target on the image is the current focus of our work. We have tested various techniques for patch positioning. These include fixed locations, regions containing a high concentration of edges, and clear and fuzzy artificial patches. These techniques are described in the following sections.

A. *Fixed Locations (*Central*,* Individuals *and* Merge*)*

This strategy has been tested using three different layout patterns: *Central*, *Individuals* and *Merge*. Each was used to identify the patches' locations. Each technique has different numbers of patches, patch sizes and patch locations. The patches' properties will help us categorise the effect on the shape of the error surfaces.

Applying different techniques on the same image will result in different error surfaces because different strategies capture different views and perform the calculation process based on different pixel selections.



(a) (b) (c)

Fig. 3: Locations of each strategy: (a) *Central*, (b) *Individuals*, (c) *Merge*

Fig. 3 shows visually where each strategy is targeting and what pixel samples it includes when applying them to images.

(a) (b) (c)

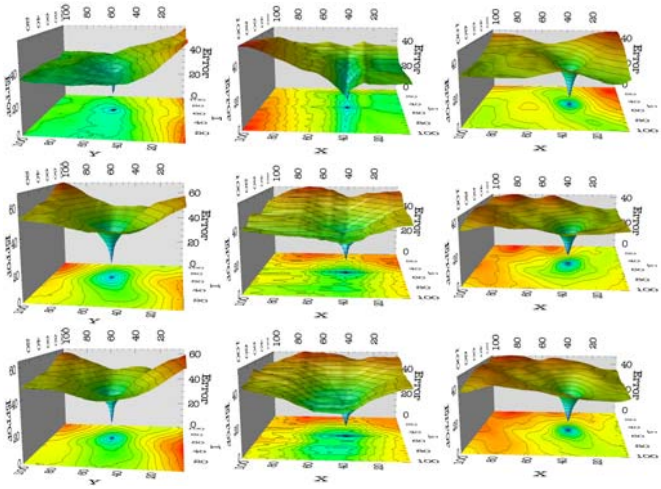
Fig. 4: Three different testing images: (a) Garden, (b) Door, (c) Flower

Examining Fig. 3(a), the *Central* strategy, shows that it is obvious that when applying it on the Garden image (Fig. 4(a)) the grass will occupy most of the patch. By looking back at the artificial images we can see that large single colour regions such as the big black square can lead to a lot of similarity and have many close minimal points. The grass is thus a poor region to target as the Garden image includes a lot of grass and is likely to yield a wide ambiguous global minimum. The situation is somewhat similar when applying the *Central* strategy to the Door image but rather better when applied to the Flower image (Figs. 4(b) and 4(c)). The *Central* patch in the door image also contains little variation in colour and there is the possibility that the error surface will be very flat in this region. This strategy worked best on the flower image, primarily because of the variety of colours captured by the *Central* strategy in the Flower image. In Fig. 3(b), the

Individuals strategy, the central box is divided into four equal size parts and distributed around the diagonals. In the Garden image this distribution is beneficial because it leads to collecting more variation and results in a smoother error surface with a well-defined minimum. Edges are important in terms of these sorts of variation and by examining the regions of the images captured by the *Individuals* boxes (Fig. 3(b)), we can see that *for these images* we tend to gather more variety of colour than with the *Central* strategy. This variety helps in producing error surfaces which descend more reliably towards the global minimum.

Fig. 3(c) shows that the situation with the *Merge* strategy is quite similar to the *Individuals* strategy except that we decreased the four patches size and included an extra patch in the centre position. The extra patch we added in the centre still contains only grass for the Garden case, but the other patches sample other colours and offset this effect.

Fig. 5 shows a comparison of some representative regions of the error surfaces for all three strategies for all three images. Each figure contains three overlapping surfaces representing the *Merge*, *Individuals* and *Central* strategies.



(a): Garden (b): Door (c): Flower

Fig. 5: Visual comparison for different error surfaces when applying different strategies on different images (top is *Central* strategy, middle is *Individuals*, bottom is *Merge*).

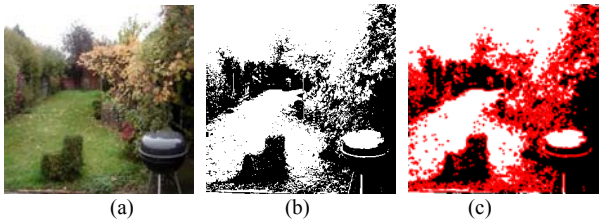
Fig. 5(a) shows the error surfaces for all three strategies when applied to the Garden image. It shows that the *Individuals* and *Merge* strategies perform very similarly. They both have better surfaces than the *Central* one (which is the lowest, flattest surface) because of the larger slope they produced. This is because the *Central* patch mainly covers grass areas, which is homogeneous in colour and widespread. The situation differs from one image to another and the shape of the error surfaces differs according to the image characteristics. Fig. 4(b) shows an image of a door with a brick wall surrounding it. It has a repeating pattern in the brickwork and a contiguous region of colour in the door itself: two features that we expect to cause problems. Local minima are likely to be generated by the brickwork and flat regions by the contiguous colour regions. Fig. 5(b) shows the error surfaces when applying all three strategies to this image: all three plots having relatively well behaved *overall* shapes, but contain local minima and ripples. The *Central* plot is less smooth than the other two (and shows some ripples and local minima), but the *Merge* and *Individuals* error surfaces are very similar. The door has repeating brickwork patterns (causing the ripples) and targeting anywhere within the (contiguously coloured) door may reduce the slope. The *Merge* and the *Individuals* patches happen to be located mostly in the corners where we have some colour variety: moving the patches in these areas generates a better error surface.

Fig. 5(c) is a comparison between the three strategies for the Flower image. The graph shows all three plots smoothly heading down and having nice shapes. The *Central* plot is slightly more rippled than the other two.

B. Edge Detection

Edges (such as those in the brickwork in the door image) play an important role which can have a significant effect on the shape of the generated error surface. There are many ways of defining edges and we define them as places where there is a sudden variation in brightness because that variation might be helpful in identifying the minima on the error surface. There are numerous algorithms to identify edges and they vary significantly in terms of their performance, speed and accuracy. These techniques have been used for different purposes such as object tracking [2] and image comparison. There are also many applications to track moving objects such as vehicles which were based on these techniques [1]. Sobel [4], Moravec [6] and Robert [7] are examples of those widely used techniques which we have tested and applied over the images to see if any algorithm will help to detect patches suitable for generating error surfaces for minimisation and stabilisation.

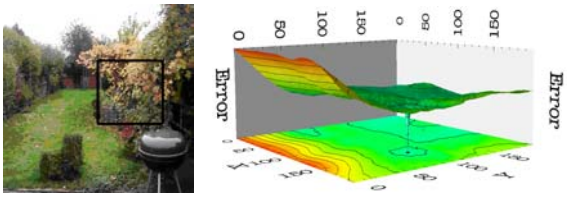
For simplicity we converted the images to binary before applying the edge detection algorithm. This will identify the edges more accurately and produces less noise [8]. Fig. 6 shows the results of the process: the grey areas in Fig. 6(c) are the edges that are detected.



(a) (b) (c)

Fig. 6: Converting the image (a) into binary (b) then detect the edges (c)

The next step is to identify the "best" patch and in our case we chose areas with the largest number of edges. The patch selected is shown in Fig. 7(a) and the resultant error surface in Fig. 7(b).



(a) (b)

Fig. 7: Patch located on region containing the most edges (a) and used as a target to draw the error surface (b)

The minimum in the error surface in Fig 7(b) is very narrow where it's heading down to the global minimum. This will cause problems when trying to find the global minimum from any large displacements in the camera's angle of view.

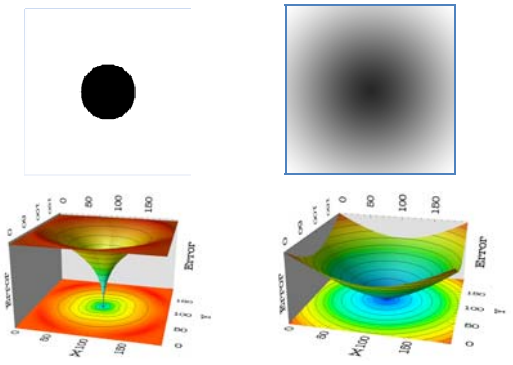
C. *Clear and Fuzzy*

Edges are an important feature in generating the shape of error surfaces. Clear images usually have hard edges whereas fuzzy ones contain soft edges.

1) *Artificial Images*

Images with clear resolution will usually have sharp edges and we expect sharp variations on the surface as patches used for generating error surfaces pass through such image regions. In non fuzzy images we expect error surface slopes to be steeper than those images with smoother variation.

Fig. 8 shows a visual comparison between two error surfaces which were generated from clear and fuzzy images.



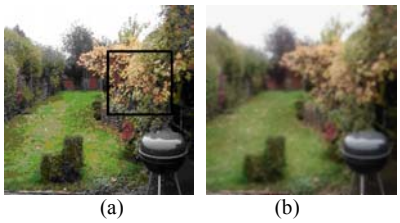
Clear Circle Fuzzy Circle

Fig. 8: Two error surfaces after running the comparison process over the clear and fuzzy images

The visual comparison shows clearly the difference between the two and the shape of the area surrounding the global minimum. Both of these surfaces have well-defined global minima and the minima could be found effectively using simple control systems such as PID. The surface generated from the clear image is steeper than the smoother surface generated by the fuzzy image. Fuzzy images will have more gradual changes which makes them a good choice if we need a smooth error surface for the control system to work on.

2) *Real World Images*

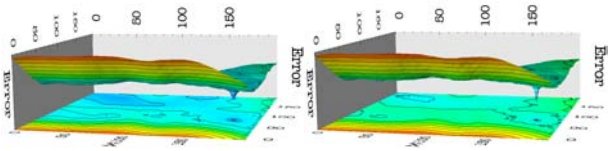
Extensive experiments were undertaken to investigate the use of edge detection for patch selection, a typical example is shown in Fig. 9(a), which contains a box indicating the area with the largest number of edges.



(a) (b)

Fig. 9: Area with highest number of edges (a) applied over the fuzzy image (b)

As in the previous synthetic images case, we fuzzified the Garden image, Fig. 9(b). Fig. 10 shows a comparison between the two surfaces obtained with the two images using the patch indicated.



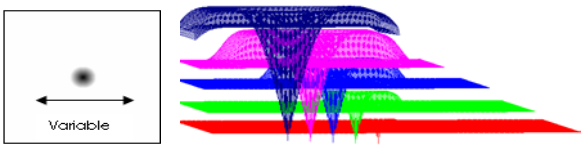
(a) (b)

Fig. 10: Comparing hard edges (b) with smooth edges (a) error surfaces

The smooth edges provide a wider minimum, as expected, although the surfaces are generally disappointingly flat.

D. *Artificial Patches*

A further experiment examined what artificial shapes produce the best error surface shapes. We use fuzzy blobs of different sizes and gradually changing colour from the centre towards the circle's edge. The patch used captures the entire fuzzy blob and some white space surrounding it. The purpose is to investigate which of those blobs produce the best error surface for minimisation. We then use that artificial patch to iterate through the real world image to find the best match between real world features and the artificial shape. This best match patch is then used as the target in the hope that the error surface will be close to what was produced in the artificial examples (Fig. 8). Fig. 11(a) shows the shaded circle in the centre which will be used to find the best error surface shape that can be used for the comparison process. We tested sizes for the fuzzy blobs varying from 10 to 50 pixels in diameter but the patch we capture is from 30 to 150 pixels diameter. The main reason is to capture a patch which includes a dark blob in the middle with some white space surrounding it. Different blob sizes produce different error surfaces with different slopes (Fig. 11(b)).



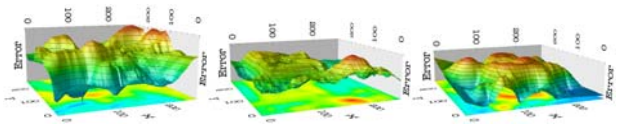
(a) (b)

Fig. 11: Error surfaces (b) corresponding to different blob sizes (a)

The flattest surface corresponds to the smallest patch with a size of 30 x 30 pixels. The other surfaces were produced with blobs of sizes 60, 90, 120 and 150 pixels. Fig. 11(b) clearly shows that increasing the blob size improves the surface from the controller's point of view. Such artificial patches can be used to find similar features in images such as the three test images. We tested it on all three images (Garden, Door and Flower) and experimented with different patch sizes to see what features were captured for the comparison process. The experiment was based on only one of the RGB colour channels (informal experiments showed no noticeable difference in performance between them). We arbitrarily chose the green channel for later experiments. The boxes indicated in Fig. 12(a-c) show the patches selected for each colour channel and after applying the comparison process between the artificial patch and the real image over the different channels. The surfaces generated are shown in Fig. 13(a-c). These results used the smallest patch size (30 x 30).

(a) (b) (c)

Fig. 12: The green channel patch in Fig. (a) is located in the top-most left box and in Fig. (b) is in the middle between the other two boxes. The green channel box in Fig. (c) is located in the top right corner where the blue and the green patches are overlapping.



(a) (b) (c)

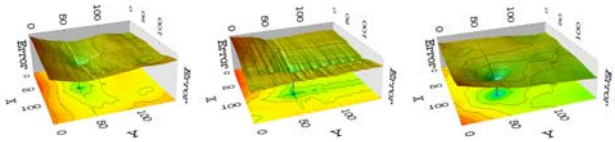
Fig. 13: Applying the 30x30 artificial patch over the three images

It is obvious that none of the error surfaces in Fig. 13 are useful. We also see visually that the green channel patches in all three images were captured in very "bland" areas with little variation which can cause the error surface to have many local minima. We now take the experiment to a further stage where we repeat the same procedures but with different artificial patch sizes. The green channel boxes drawn in the Figs. 14(a), 14(b) and 14(c) indicate the best patch found after applying the comparison between the artificial patch with the real image over that channel only. We have used an artificial patch size of 150 x 150 pixels to find the best patch possible.

(a) (b) (c)

Fig. 14: The green and the blue channel boxes in Fig. (a) and Fig. (b) are mostly overlapping; in Fig. (a) it is the left-most box and in Fig. (b) it is the top-most box. The green channel box in Fig. (c) is overlapping with the red channel box and is the top-most box in the image.

We now apply the comparison with different artificial patch sizes to see how the error surface can differ when capturing a bigger patch and apply the process using a large patch size.



(a) (b) (c)

Fig. 15: Applying the 150x150 artificial patch over the three images

The difference between the patch sizes is clear and dramatic. For instance, if we compare the Door error surface (Fig 13(b)) generated by the 30 x 30 patch with the error surface generated by the 150 x 150 patch (Fig 15(b)), we can see how the large patch removed a lot of noise on the surface and made it much smoother and easier for the controller to navigate. A similar evaluation also applies for the other two error surfaces where the patch's enlargement made a significant difference for both Garden and Flower images.

V. Whole Image Sampling Strategies

We cannot in general predict what image properties we will encounter and where to apply which strategy to capture the best patch possible. We therefore applied some "whole image strategies" which ensure that no matter what features and what colours the image includes, we still target the entire image and use the entire image as a patch. Using all the pixels requires a prohibitively large amount of processing time and therefore, we applied different ways of capturing limited number of pixels but distributed around the image. There are two ways used to distribute the pixels in the image, one is static positioning and the other is random positioning.

A. Static Positioning

This strategy uses most of the image space to capture a variety of pixels from the entire image. The static positioning method uses a fixed grid to select which pixels are targeted. There are many ways of targeting the pixels

but we attempted to define a layout to ensure that on all translations some sampled pixels will overlap. Selecting based on a regular grid does not achieve this: if we sample every *n* pixels we will not get any overlap if we displace the image by *(n-1)* pixels.

We resolved this issue by defining a simple way to guarantee some overlap on every movement and therefore we made the selection process to take place every 10, then 9, then 8, etc., down to a spacing of 1. This process of selecting the pixels is repeated right across the image. Fig. 16 shows the selected pixels in the *Static Positioning* strategy. We will be using the above indicated locations as our patch for the comparison process.

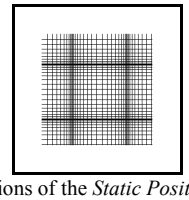
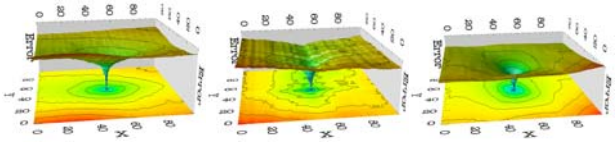


Fig. 16: Locations of the *Static Positioning* strategy

We initially ran the comparison process for each colour channel separately and the result was that the error surfaces for them all was quite similar and therefore we decided to use only a single colour channel for the comparison process (again we chose green for no particular reason). Fig. 17 shows error surfaces for the Garden, Door and Flower images using static positioning over the green channel only.

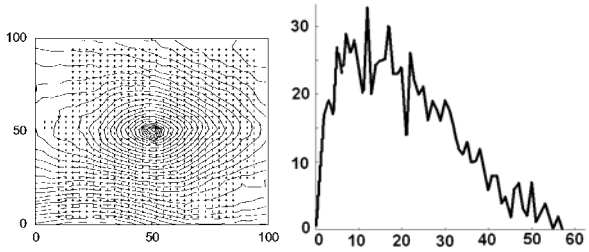


(a): Garden (b): Door (c): Flower

Fig. 17: Error surfaces for different images after applying the *Static Positioning* strategy

It is clear that all three error surfaces have a smooth slope down towards their minimum and the noise and local minimal are almost completely eliminated (although local minima due to the brickwork texture are apparent in Fig. 17(b)), which is potentially valuable from the controller's point of view. The next stage is to test how well the controller navigates over these surfaces.

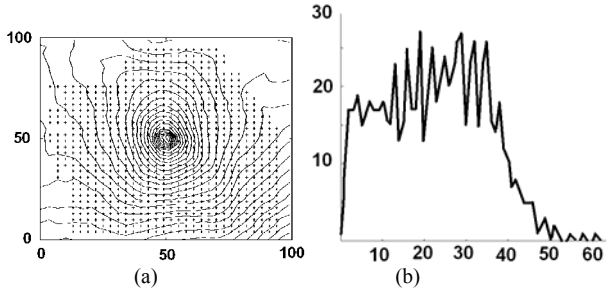
Running the control algorithm described above from all starting points on the error surface will give us the ultimate result which verifies from what parts of the surface we can reach the target and get ourselves back to the original location. Fig. 18 is produced from Garden's green channel error surface. Fig. 18(a) shows a number of dots on the contour map which indicate the locations from which the minimum was successfully reached using the simple control algorithm. Fig. 18(b) shows the distribution of the number of steps required to reach the global minimum from all the successfully minimised starting locations.



(a) (b)

Fig. 18: Successful areas on the contour map for the Garden image

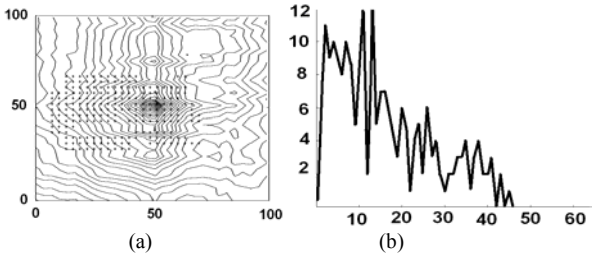
The statistics show that 77% of the positions on the image were successful. Fig. 19 shows the number and locations of the successful starting points on the error surface for the Flower image.



(a) (b)

Fig. 19: Successful areas on the contour map for the Flower image

73% of the entire image's locations were successful. Fig. 20 shows the same data for the Door image, for which 20% of the starting locations succeeded.



(a) (b)

Fig. 20: Successful areas on the contour map for the Door image

This poor success for the Door image is due to the homogeneous areas and repeating patterns in the image.

B. *Random Positioning*

Random Positioning of pixels is another technique which we used to target the entire image rather than positioning a patch(s). Fig. 21 shows the uniform random distribution of locations used in this strategy.

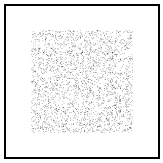
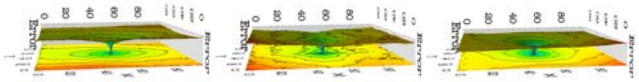


Fig. 21: Distributions of the pixel locations using the *Random Positioning*

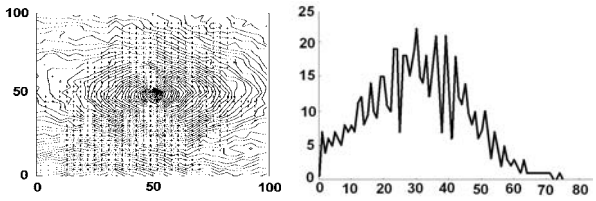
In this approach we also ran the comparison process over each of the colour's channels separately to see if there was any difference amongst any of the produced surfaces. As in previous cases, we have chosen the green channel to base our experiments on as there was no noticeable difference.



(a): Garden (b): Door (c): Flower

Fig. 22: Error surfaces after applying the *Random Positioning*

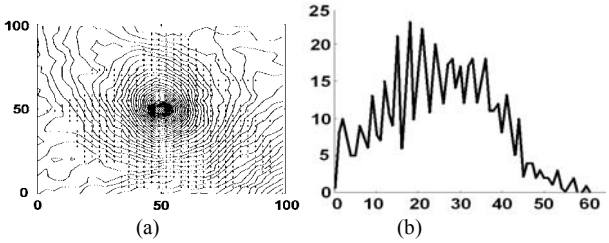
Fig. 22 shows the error surfaces generated using the random positioning strategy based on green channel only. The error surfaces are quite similar to the surfaces generated by the Static Positioning strategy. This similarity comes from the way we generated the surfaces where in both cases we targeted the entire image to be the patch for the comparison process. Fig. 23(a) is a contour map for the Garden image with the indication of the successful points and the Fig. 23(b) is the graph showing how many points took how many steps in order to achieve the process.



(a) (b)

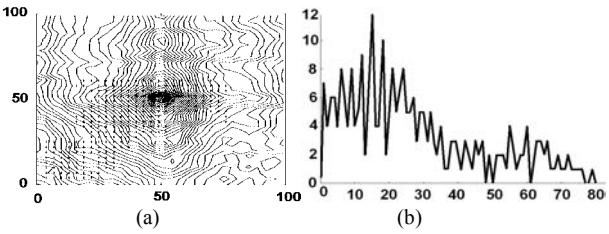
Fig. 23: Successful areas on the contour map over the Garden image

In that case, using only the green channel, 59% of the starting locations succeeded using the *Random Positioning* strategy. Fig. 24(a) shows the contour map for the Flower image with the indicated points showing the successful area, which is 52% of the image locations.



(a) (b)

Fig. 24: Successful areas on the contour map for the Flower image



(a) (b)

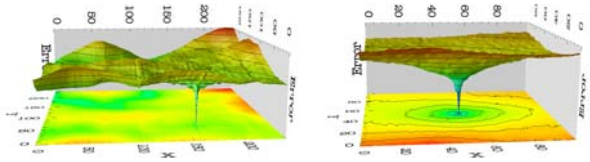
Fig. 25: Successful areas on the contour map for the Door image

Finally we apply the same technique to the Door image. Fig. 25(a) shows the successful area covering 24% of the image: a slight improvement over the *Static Positioning*.

VI. CRITIQUE

The purpose of choosing different strategies was to implement a solution for the best patch found within the image. The fixed locations strategies perform well only if the patches happen to capture some good features in order to build the error surface but this is unacceptably dependent on the properties of each individual image. The patch size is also a major consideration in the shapes of the error surfaces: larger patches tend to produce better error surfaces and the effects of "noise" decrease. The strategies using edge detection and artificial patches also performed unreliably in general.

There is a dramatic change in performance between *patch positioning* and *whole image sampling* strategies. For example Fig. 26(a) was generated by applying the 60x60 *Artificial Patch* to find the best positioning before applying the comparison process and the surface in Fig. 26(b) was generated by applying the *Random Positioning* strategy. The random strategy uses fewer pixels for the comparison process and the difference between the two is clear: random surface is smooth, evenly sloping and has a well-defined single global minimum whereas the artificial patch surface is noisy and has multiple local minima. Both static and random positioning strategies perform well and quite similarly by targeting pixels distributed over the entire image.

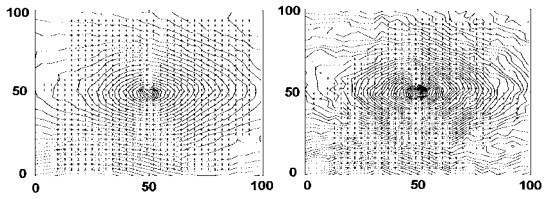


(a) (b)

Fig. 26: *Patch Positioning* (a) and *Random Positioning* (b) surfaces

Both whole image sampling strategies are effective although in these experiments more pixels were sampled in the static sampling strategy. Despite the large number of pixels the error surface contains a number of regions from which our simple controller was unable to minimise. This may be partly due to the simplistic nature of the controller, but in general the randomly positioned pixels yield steeper and deeper global minima: compare Figs. 27(a) and (b).

In general the *Random Positioning* strategy does not suffer from regions in which our controller was unable to minimise and therefore seems to be a better solution.



(a) Static (b) Random

Fig. 27: Successful areas on *Static* (a) and *Random Positioning* (b)

The minimisation method used on the surfaces generated by this work was essentially a P controller moving the target location iteratively across the surface. The controller used information only from the local slope of the error surface which was assessed using three points to obtain the direction of the slope; this minimises the computational load by only requiring the sample pixels to be compared three times between each movement of the actuators. A summary of the numbers of pixels sampled to generate some error surfaces can be seen in Table 1.

Figure		
5	10,000 (*Central* + *Individuals*)	10,580 (*Merge*)
7	10,000 (edge detection)	
10	10,000 (artificial patches)	
13	900 (small artificial patches)	
15	22,500 (large artificial patches)	
17	14,600 (static sampling)	
22	2,500 (random sampling)	

Table 1: Number of pixels sampled for various surfaces

In order to control the physical platform a more refined and sophisticated system is likely be required. In addition a controller that takes advantage of the nature of the kite platform's typical motion will assist in improving performance. For example the platform is suspended on strings (using a "picavet" arrangement) beneath the kite string itself and thus the main mode of motion is as a pendulum with a simple harmonic motion. Thus the use of information about the platform's trajectory in previous iterations could help in predicting the location of the minimum on the error surface. The picavet arrangement of strings also minimises rotation of the platform which (along with scaling) is ignored in this work. Rotation of the image due to rotation of the platform and scaling of the image due to changes in altitude are both factors which can be addressed in similar ways to the translations considered here although this will increase the computational load.

VII. Conclusion

The ultimate conclusion behind all the patch positioning strategies and the applied experiments is that wider distributions of pixels will capture more of the variation in real images and therefore will produce better error surfaces and less noise. In order to achieve this some form of whole image sampling is the best technique. This leads to successful navigation on the error surface generated by the entire image and not depending on a specific feature, shape, colour, etc. To design a successful visual stabilisation control system for our Intelligent Kite Aerial Photography Platform (iKAPP) system we need a strategy which reliably generates a good error surface where the controller will have the opportunity to get us back to our target from most positions in most images. In addition the processing load is important as the platform uses a fairly low specification: 1GHz processor with 512Mb of RAM. Fast processing will allow us to process images at a higher frequency and therefore move the actuators at a higher frequency which will improve the stabilisation performance. Every error surface has a certain region within which the controller can succeed and if the kite system encounters a lot of turbulence then, more images and faster processing will help to keep the camera within this region.

VIII. Future Work

We are currently working on refining the pixel sampling strategy with the main aim of ascertaining how few pixels will still produce a reliable error surface for minimisation. The next step will be the deployment of the complete control system on the platform under laboratory conditions where we will use VICON motion capture equipment to assess the motion of the platform and response rates of the complete system. Once this work is complete and we have determined whether it is possible to obtain performance sufficient for useful stabilisation for translations we will deploy the system in the field and move on to examine the problems of rotation and scaling due to changes in altitude of the platform under real world conditions.

REFERENCES

[1] A. Behrad, A. Shahrokni, and S. Ahmad Motamedi, (2001), "A robust vision-based moving target detection and tracking system", In *Proceeding of Image and Vision Computing Conference*, University of Otago, Dunedin, New Zealand.

[2] B. Jung and G. S. Sukhatme, (2004), "Detecting Moving Objects using a Single Camera on a Mobile Robot in an Outdoor Environment". In *proceeding of International Conference on Intelligent Autonomous Systems,* pp. 980--987.

[3] D. Sellers, (2001), "An Overview of Proportional plus Integral plus Derivative Control and Suggestions for Its Successful Application and Implementation", In *Proceedings of the International Conference for Enhanced Building Operations.*

[4] R. Duda and P. Hart, (1973), "Pattern Classification and Scene Analysis", John Wiley and Sons,'73, pp271-2.

[5] F. Labrosse, (2006), "The visual compass: performance and limitations of an appearance-based method", *Journal of Field Robotics*, 23(10), pages 913-941.

[6] H. Moravec, (1980), "Obstacle Avoidance and Navigation in the Real World by a Seeing Robot Rover," *tech. report CMU-RI-TR-80-03, Robotics Institute, Carnegie Mellon University & doctoral dissertation, Stanford University*, September.

[7] L. Roberts, (1965), *"Perception of 3-D Solids*, Optical and Electro-optical Information Processing", MIT Press.

[8] N. Efford, (2000), "Digital Image Processing", Addison-Wesley Press, ISBN: 0-201-59623-7